Computer Communications and Networks (COMN)
2014/15, Semester 2

Assignment Part 2 Results Sheet

Forename and Surname:	
Matriculation Number:	s1210313

Question 1 – Experimentation with Go-Back-N:

	Throughput (Kilobytes per second)		
Window Size	**Delay = 10ms**	**Delay = 100ms**	**Delay = 300ms**
1	10.3 | 87.638KB/s	79s | 5.007KB/s	533s | 1.685KB/s
2	10 | 88.153KB/s	92s | 9.770KB/s	267s | 3.358KB/s
4	5s | 152.582KB/s	47s | 18.937KB/s	133s | 6.720KB/s
8	4s | 209.645KB/s	23s | 38.165KB/s	67s | 13.25 KB/s
16	7s | 106.967KB/s	16s | 54.140KB/s	44s | 20.094KB/s
32	18s | 49.845KB/s	18s | 47.592KB/s	62s | 14.351 KB/s
64	81s | 11.007KB/s	24s | 36.990KB/s	81s |10.977KB/s
128	207s | 4.339KB/s	188s | 4.767KB/s	119s | 7.512KB/s
256	337s | 26.60KB/s	459s | 19.54KB/s	343s | 2.619KB/s

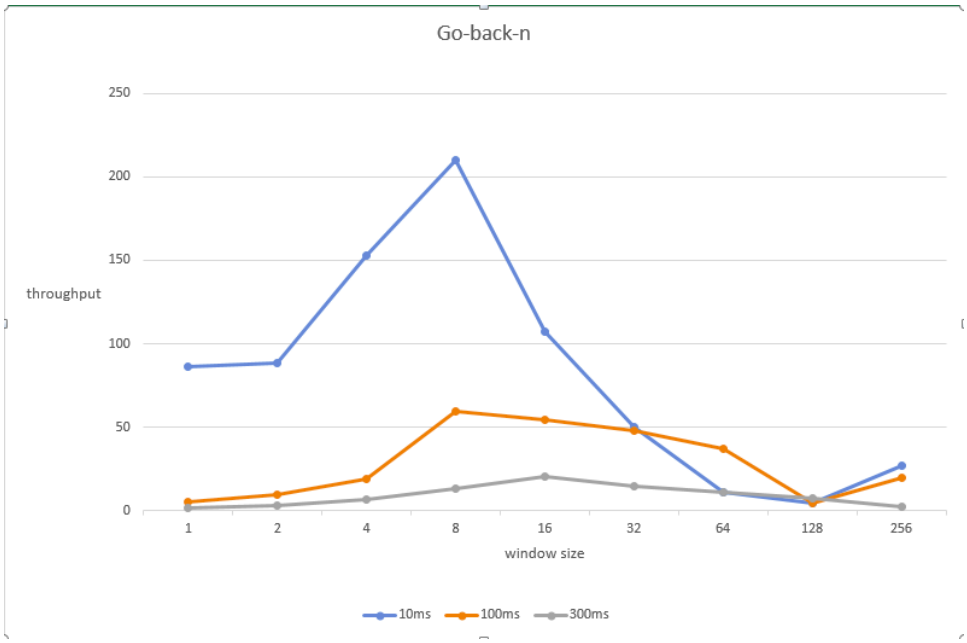
Retry : 20, 50, 150

I anticipated that the optimal Retransmission Timeout would be slightly greater than the Response Time, or the propagation delay, plus some constant file transmission rate. For instance, if the best performance was at 40 milliseconds on the previous assignment, that would mean that

RetransmissionTime = ResponseTime = 2 * PropagationDelay + FileTransmissionRate
40 = 20 + FileTransmissionRate
FileTransmissionRate = 20

Given this, the optimal retransmission time should be 40, 220, and 620. Any lower and packets will be retransmitted before they are acknowledged, reducing throughput. Despite this, from experimentation, lower retry timeouts increased the throughput

Create a graph as shown below using the results from the above table:



Question 2 – Discuss your results from Question 1.

Go-Back-N may have up to N unacked packets in the pipeline at a time. When a ack is received, it is considered cumulative---that all the packets before it have also been acknowledged. The sender maintains a timer for the oldest packet in the pipeline, and when that expires all unacked packets are re-transmitted.

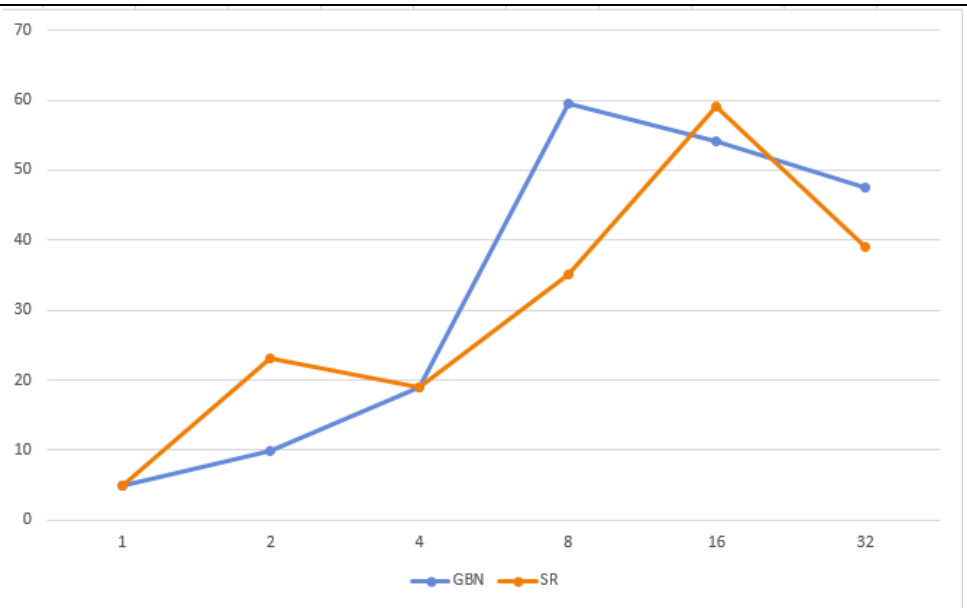
Because of this, when choosing a window it's a balance between taking advantage of any delays in the system, and being cognizant of the packet-loss in the system. For instance, with a 100ms propagation delay, window sizes 1,2, and 4 perform poorly: they don't take advantage of the fact that the sender will have to wait at least the RTT to receive a receipt of a package. Conversely, with a small propagation delay of 10ms, when the sender has a larger window, it's performance plummets: it must resend all 128 packets in the window if there's a packet dropped at any point.

Question 3 – Experimentation with Selective Repeat

	Throughput (Kilobytes per second)
Window Size	**Delay = 100ms**
1	182s| 4.913
2	38s | 23.428
4	46s | 19.146
8	25s | 35.362
16	15s | 58.950
32	22s | 39.136

Retry-timeout is the same as the previous question. Note, I introduced a bug just before hand-in, so the 2b receiver no longer functions.

Question 4 - Compare the throughput obtained when using "Selective Repeat" with the corresponding results you got from the "Go Back N" experiment and explain the reasons behind any differences.



The purpose of GBN is to fill the pipe. By sending a window full of packets, GBN avoids the utilization issues that were encountered with stop and wait. However, when the window size and the propagation delay are both large, GBN exposes itself to many retransmissions. When there are many packets in the pipeline, if a single packet is dropped, the entirety of the window must be retransmitted. This fills the pipeline with retransmissions which are unnecessary---as it's likely some of those retransmitted have already been received.

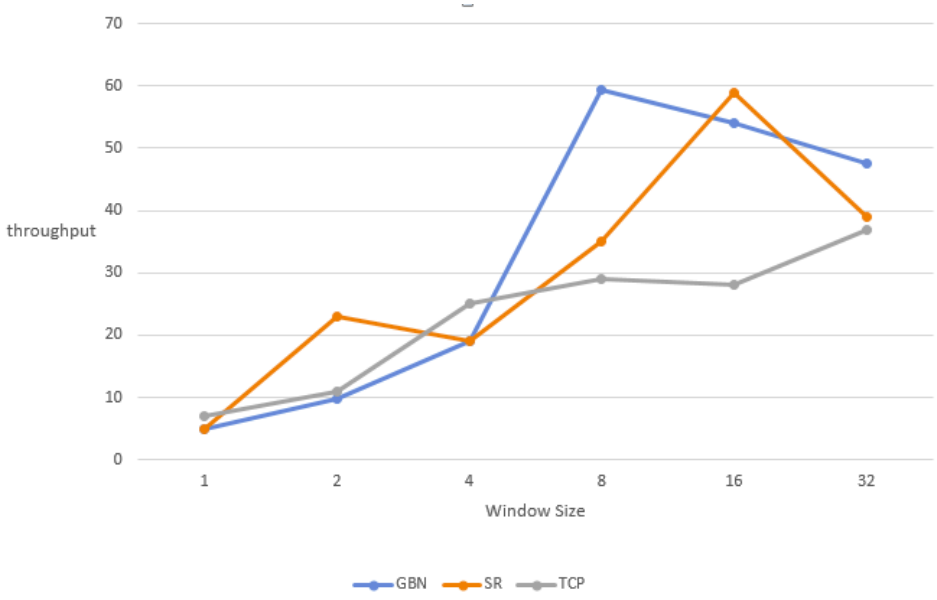
SR achieves better performance over larger windows and delays by allowing packets to be acknowledged out of order. Instead of retransmitting all of the packets in the window SR retransmits only the unacknowledged packets. This allows us to better utilize the pipeline and prevents it from filling with retransmission, improving through put.

Given this, it's unexpected that there would be a larger than 10 kilobyte difference between the throughputs of GBN and SR at window size of 2, as the window size isn't large enough for a gain in the retransmission; and it would be anticipated that SR would outperform GBN at a window size of 32, as SR would be more efficient at retransmitting, allowing for better throughput.

Question 5 – Experimentation with *iperf*

	Throughput (Kilobytes per second)
Window Size (KB)	**Delay = 100ms**
1	55kbits/s | 7 kbytes/s
2	95kbits/s | 11 kbytes/s
4	197kbits/s | 25 bytes/s
8	239kbits/s | 29 kbyte/s
16	227 kbits/s | 28 kbytes/s
32	298 kbits/s | 37 kbytes/s

Question 6 - Compare the throughput obtained when using "Selective Repeat" and "Go Back N" with the corresponding results you got from the *iperf* experiment and explain the reasons behind any differences.



Rather than defining the Retransmission Timout and holding it constant throughout the whole of the transmission ---as GBN and SR do---TCP samples transmissions and uses the weighted average estimate of the RTT to adapt the timer

as the pipeline's congestion changes. Alongside this, TCP has a fast-transmit; when three duplicate acks are detected the sender resends the oldest unacked segment, as it's likely that it's been lost. In short, TCP has flow control which prevents the receiver from being overwhelmed.

TCP shares many of the traits of GBN and SR. TCP does not individually ACK received out of ordered segments. However, like SR, TCP---depending on the implementation---will buffer packets received out of order. As a result its performance, save for at a window size of 16, is similar to both SR and GBN.
The average throughput of a TCP connection is :(1.22 * MSS)/ (RTT * sqrt(L))
For this assignment it would be (1.22*1000)/(200*sqrt*(0.01)